



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/677,081	09/30/2003	Garrett Herschleb	884.A78US1	4055

21186 7590 09/21/2006

SCHWEGMAN, LUNDBERG, WOESSNER & KLUTH, P.A.
P.O. BOX 2938
MINNEAPOLIS, MN 55402

EXAMINER

VU, TUAN A

ART UNIT PAPER NUMBER

2193

DATE MAILED: 09/21/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/677,081	HERSCHLEB, GARRETT	
	Examiner	Art Unit	
	Tuan A. Vu	2193	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 30 September 2003.
- 2a) ☐ This action is **FINAL**. 2b) ☒ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 1-30 is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) ☐ Claim(s) _____ is/are allowed.
- 6) ☒ Claim(s) 1-30 is/are rejected.
- 7) ☐ Claim(s) _____ is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☒ The drawing(s) filed on 9/30/06 is/are: a) ☐ accepted or b) ☒ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some * c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
2. ☐ Certified copies of the priority documents have been received in Application No. _____.
3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).
- * See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- | | |
|--|---|
| 1) ☒ Notice of References Cited (PTO-892) | 4) ☐ Interview Summary (PTO-413) |
| 2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948) | Paper No(s)/Mail Date. _____ |
| 3) ☐ Information Disclosure Statement(s) (PTO/SB/08) | 5) ☐ Notice of Informal Patent Application |
| Paper No(s)/Mail Date _____ | 6) ☐ Other: _____ |

DETAILED ACTION

1. This action is responsive to the application filed 9/30/2003.

Claims 1-30 have been submitted for examination.

Drawings

2. New corrected drawings in compliance with 37 CFR 1.121(d) are required in this application because the current drawings are substantially made out of informal scribbling with a pen. Applicant is advised to employ the services of a competent patent draftsman outside the Office, as the U.S. Patent and Trademark Office no longer prepares new drawings. The corrected drawings are required in reply to the Office action to avoid abandonment of the application. The requirement for corrected drawings will not be held in abeyance.

Claim Objections

3. Claim 15, 19 are objected to because of the following informalities:

Specifically, there appears to be a missing term between 'cause the variable' and 'loaded' (line 5). Further the use of 'passage of a parameter', in light of the known concept of parameter passing, needs to be readjusted because the parameter is not actively passing but subject to a process that passes it, the process actually doing the passing; that is, 'passage of' should be 'passing of'.

Claim 19 for reciting 'passage of parameter' needs also be corrected.

Appropriate correction is required.

Claim Rejections - 35 USC § 101

4. 35 U.S.C. 101 reads as follows:

Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

Art Unit: 2193

5. Claims 1-6, 15-18, 22-26 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

The Federal Circuit has recently applied the practical application test in determining whether the claimed subject matter is statutory under 35 U.S.C. § 101. The practical application test requires that a “useful, concrete, and tangible result” be accomplished. An “abstract idea” when practically applied is eligible for a patent. As a consequence, an invention, which is eligible for patenting under 35 U.S.C. § 101, is in the “useful arts” when it is a machine, manufacture, process or composition of matter, which produces a concrete, tangible, and useful result. The test for practical application is thus to determine whether the claimed invention produces a “useful, concrete and tangible result”.

The method of claim 1 recites executing first and second module, each including instructions that causes variable loading, parameter passing, or parameter accessing using offset operation. As a whole, the method amounts to internal working of computer execution and there is no reasonable conveying that by obtaining the result of this execution, some application is perceived as yielding a real-world concrete result, in terms of it being tangible, and useful. That is, the claim as a whole, for merely reciting that some module is being executed without providing what would benefit from such execution so to lead to a result as required from the Practical Application Test, does not amount to a internal abstract computer data, not a real world useful result. Therefore claim 1 and claims 2-6 are rejected for leading to a non-statutory subject matter.

Claims 22-26 amount to computer instructions stored in a media to execute instructions in the same manner as claims 1-6; hence fail to provide a tangible result based on such execution when there is insufficient teaching as to how this execution would interact with any application for users to benefit of its result, particularly when there is no result being reasonably conveyed from the claims as a whole as being concrete, useful and tangible, absent any form of application via which a user can perceive such result.

Claim 15 recites a system comprising a compiler. This compiler is disclosed (refer to Specifications pg. 7, first line) as being software application, hence the claimed system amounts to just a software entity having some function, and so without any hardware support to realize such functionality. The claim as a whole cannot be construed as able to generate real-world result when the software is not disclosed as being executed via computer tangible support.

Claims 15-18 together fail to provide a result as required by the Practical Application Test, and are rejected for leading to a non-statutory subject matter.

Claim Rejections - 35 USC § 112

6. The following is a quotation of the first paragraph of 35 U.S.C. 112:

The specification shall contain a written description of the invention, and of the manner and process of making and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the same and shall set forth the best mode contemplated by the inventor of carrying out his invention.

7. Claims 3, 17, 24 are rejected under 35 U.S.C. 112, first paragraph, as failing to comply with the written description requirement. The claim(s) contains subject matter which was not described in the specification in such a way as to reasonably convey to one skilled in the relevant art that the inventor(s), at the time the application was filed, had possession of the claimed invention.

Claim 3 recites parameter from claim 1 not pushed into the stack 'as part of the call from first module to the second module'. From the Specifications, Figure 4 shows procedure A as having parameter passed by reference, and these parameters appear to be on the stack as part of the calling stack using the Base Pointer and the return address, all of which being part of Figure 6B wherein procedure A is being called (line 674) and variables needed for this function call are pushed on the stack (see pg. 15 block 802). Therefore, as part of this calling instruction, line

Art Unit: 2193

674, Fig. 6B, the local variables are pushed into the stack and this is construed as being known concept from the prior art of Figure 1 and 2. Hence, it appears that the variables are to be pushed in order for them to be part of the calling instruction, in regard to which a same calling instruction form is depicted in line 110 of Figure 1 and line 414 of Figure 4. One skill in the art would not be able to construe this limitation based on what the Disclosure description of the calling of A (Fig. 1, Fig. 6A, 6B, 7); and the claim is rejected for failing to provide appropriate description to convey that the inventor has possession of this specific limitation.

The limitation will be treated as though the parameters are pushed into the stack to support the passing by reference (i.e. as part of the calling of A) but not pushed any more at runtime where a pop instruction would be needed; as shown in Figure 1 or Figure 4 of the Specifications.

Claims 17 and 24 recite the limitation of claim 3 and are rejected for the same rationale.

8. The following is a quotation of the second paragraph of 35 U.S.C. 112:

The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

9. Claims 7-21, 27-29 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

Claim 7 recites compiled instructions do not include instruction to load the variable onto a stack, during execution, based on the call instruction (last paragraph). It is noted that stack is dynamic structure to save address values of runtime call, i.e. stack for push/pop operations during execution; and based on the teaching of Figure 1 and Figure 4, values of parameters are seen as being pushed into the stack for supporting the call by reference as described in the claim.

Art Unit: 2193

Based on the calling of a function in which parameters (see Specifications: function A in Figure 1 & Figure 4) as mentioned above in the rejection using USC 112, 1st paragraph, it is unclear as to how not including loading of the recited variable – variable as being passed by reference -- has been effected during stack runtime. One skill in the art would not be able to make use of the claimed invention in light of the stack being loaded with the very variables such that it would become part of the calling of function A (see above). The limitation is rejected for indefinite teaching as to how the subject matter as recited enables a clear and non-contradictory understanding.

Claim 19 and 27 recite this ‘do not include ... load the variable’ limitation; thus, are likewise rejected for reciting a limitation that render the teaching of the claimed subject matter indefinite, contradictory, unsupported, inconsistent with the Disclosure, hence not setting a clear metes-and-bounds teaching for one skill in the art to make use of the subject matter.

The limitation will be treated as though the variables are in the stack based on the call of the function.

Claim 15 recites the limitation "the variable" in line 5 and would be treated as any variable. There is insufficient antecedent basis for this limitation in the claim.

Claims 8-14, 16-18, 20-21, 28-29 do not appear to cure the deficiency of the respective base claims; and are also rejected.

Claim Rejections - 35 USC § 102

10. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

Art Unit: 2193

(a) the invention was known or used by others in this country, or patented or described in a printed publication in this or a foreign country, before the invention thereof by the applicant for a patent.

11. Claim 1-5, 7-11, 13-14, 19-25, 27-30 are rejected under 35 U.S.C. 102(a) as being anticipated by Tomasz Muldner, 'C++ Programming with Design Pattern Revealed', <<http://www.cs.helsinki.fi/u/vihavain/s03/cpp/Muldner/c++Ch2.ppt>> Addison Wesley, Copyright 2002, pp. 1-42 (hereinafter Muldner)

As per claim 1, Muldner discloses a method comprising executing an executable program that comprises:

executing a first module (Main Function , pg. 1), wherein the first module includes a second instruction that calls a second module (e.g. **Pass by Reference**: swap (int &x, int &y) – pg. 16) that passes a parameter by reference, wherein the parameter passed by reference is an address of the variable; and

Muldner does not show that the first module includes a first instruction that causes a variable to be loaded onto a stack executing the second module that includes a third instruction that accesses the parameter on the stack based on an offset from a common reference for the second module; but the concept of loading into a stack an address location of a function being called and using it as a base pointer or base index to add more offset to other stack address during effecting the call has been known in the C/C++ programming (see Admitted Prior Art or **APA**: See Specifications: Figure 2). Hence, this loading of a base pointer from the calling function (e.g. a Main Function) to a subroutine (see swap (&x, &y) so that the function being called use such BP to retrieve the value of x or y in a deference process is also disclosed as being integral to the Admitted C/C++ programming language.

As per claim 2, Muldner by virtue of the rationale in claim 1, discloses executing the second module that includes the third instruction that accesses the parameter on the stack based on the offset from a base pointer for the second module (see Admitted Prior Art: Specifications, Figure 2).

As per claim 3, Muldner discloses by virtue of claim 1 and APA (see Specifications, Figure 1), wherein the parameter is not pushed onto the stack as part of the call from the first module to the second module (parameter x, and y being pushed earlier –see APA: Specifications, Fig. 1) reads on not being pushed again during effecting of Muldner's calling of *swap* (**Pass by Reference**: *swap* (int &x, int &y) – pg. 16).

As per claim 4, see Muldner for the first instruction that causes the variable to be loaded onto the stack comprises executing the first module that includes the first instruction to define the variable (see APA: Specifications: Fig. 2, 2nd para, pg. 2) or Muldner(Pass by Reference: int i=1; int j =2, pg. 16) .

As per claim 5, see Muldner (*swap* (int &x, int &y) – pg. 16) wherein the first module calls the second module through a third module, wherein a first calling sequence comprises the first module, the third module and the second module (Note: any submodule being called by another function in a C/C++ Main program reads on sequence involving first, second and third modules called in sequence as recited).

As per claim 7, Muldner discloses a method comprising: receiving a program code that includes

a first calling function having a call instruction to call a first called function that passes, by reference (*swap* (int &x, int &y) – pg. 16), a variable as a parameter of a number of

Art Unit: 2193

parameters (Note: passing an address that needs to be dereferenced reads on starting parameter leading to other parameters OR see Specifications: Figure 1, *BP* – Note: loading stack with a subroutine base pointer as by APA reads on Muldner C/C++ language with using of such base pointer parameter as parameter representing starting point for subsequent parameter fetching); and

compiling the program code to generate compiled instructions, wherein the compiled instructions do not include a compiled instruction to load the variable onto a call stack, during execution, based on the call instruction (re rationale in claim 3).

As per claims 8-10, Muldner discloses wherein the first calling function has a declaration instruction to allocate the variable, wherein the compiled instructions are to cause the variable to be loaded onto the call stack, during execution, based on the declaration instruction (see APA: Specifications: *x, y, z* - Fig. 2, 2nd para, pg. 2) or Muldner (Pass by Reference: *int i=1; int j =2*, pg. 16); wherein the called function includes an access instruction, wherein the compiled instructions are to cause the variable to be accessed from the call stack, during execution, based on the access instruction (Note: *swap (int &x, int &y* based on BP of APA – Specifications Fig. 2 -- reads on cause to access from the stack during execution by the called function), wherein the compiled instructions are to cause, during execution, the variable to be accessed using a base pointer (see APA: BP, Fig. 2) associated with the called function.

As per claim 11, refer to rationale of claim 5.

As per claim 13, Muldner discloses wherein an order of the number of parameters passed by the first calling function equals an order of the number of parameters passed by the second

Art Unit: 2193

calling function (see Passed by Reference, pg. 16 – Note: the number of dereferencing by the called function based on the reference being passed by the calling functions reads on the above).

As per claim 14, in light of the rationale as set forth in claim 12, the limitation as to cause an order of the number of parameters on the call stack for the first calling function include parameters for the first called function and parameters for a second called function and wherein an order of the number of parameters on the call stack for the second calling function include parameters for the first called function and parameters for a third called function, wherein a size of the parameters for the third called function are not greater than a size of the parameters for the second called function would have been disclosed, in light of the necessary mapping of parameters from one calling function to the next calling function to the called function; because otherwise a mismatch would have resulted in an exception by the C/C++ at runtime or linking time.

As per claim 19, Muldner discloses a system having subject matter corresponding to that of claim 7, comprising a secondary storage to store at least a part of a source code program, wherein

the source code program includes a first subroutine that includes a first instruction to define a variable and a second instruction to call a second subroutine, wherein the call to the second subroutine includes passage of the variable by reference; a dynamic random access memory to store at least a part of a call stack; and

a processor to compile the source code program to generate assembly code instructions, wherein the assembly code instructions do not cause the variable to be pushed onto the call stack based on the second instruction;

all of which limitations having been addressed in claim 7, the source code storage prior to compilation reads on secondary storage; and the dynamic RAM storage reads on runtime using stack (see APA Fig. 2 in light of the passing by reference in claim 1)

As per claims 20-21, refer to claims 8-9.

As per claims 22-25, these claims correspond to claims 1-5, hence will incorporate the corresponding rejections, respectively.

As per claims 27-30, these claims correspond to claims 7-10, hence will incorporate the corresponding rejections, respectively.

Claim Rejections - 35 USC § 103

12. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

13. Claims 6, 12, 15-18, and 26 are rejected under 35 U.S.C. 103(a) as being unpatentable over Muldner, 'C++ Programming with Design Pattern Revealed', in view of R. Stallman, GNU CC, 'Using and Porting GNC CC', <http://web.umn.edu/~gnudoc/split/egcs/gcc_toc.html> – copyright 1988-1996 (hereinafter GNUCC - Part 1: **Target Description Macros** pp. 1-40; Part 2: **Defining the Output Assembler Language**, pp. 51-66).

As per claim 6, Muldner does not explicitly disclose that wherein a second calling sequence comprises the first module, a fourth module and the second module, wherein executing the first module comprises padding the stack such that the number of entries on the call stack are

Art Unit: 2193

the same for the first calling sequence and the second calling sequence. But C/C++ language includes freeing memory in view of leakage; and one example is disclosed in the GNU CC compiler wherein padding of extra unused stack memory is disclosed via a particular compiler macro routine (see Part 1, **Passing Arguments in Registers**: `FUNCTION_ARG_PADDING (mode, type)` – pg. 30). Based on Muldner's C/C++ usage, it would have been obvious for one skill in the art at the time the invention was made to pad unused stack memory based on passing by reference (as opposed to passing by value) paradigm wherein space saving is suggested, because unused space as purported by C/C++ programming would be much enhanced when used with compiler such as GNU with compiler macro capability to provide such padding when unused space left by function calls is detected by GNUCC macro as set forth above.

As per claim 12, the subject matter having C/C++ compiler with enabling compiled instructions as having a compiled instructions to pad the call stack such that the number of entries on the call stack are the same for the first calling sequence and the second calling sequence falls under the ambit of the subject matter of claim 6; hence is rejected with the rationale as set forth therein.

As per claim 15, Muldner discloses an apparatus comprising to generate code instructions wherein the instructions comprise a first procedure and a second procedure, wherein the first procedure includes a first assembly code instruction to cause the variable to loaded onto a call stack, wherein the first procedure instruction to cause the second procedure to be called, which includes passage of a parameter by reference, wherein the parameter passed by reference is an address of the variable (e.g. Main Function , pg. 1; **Pass by Reference**: `swap (int &x, int &y)` – pg. 16), the second procedure to include a third assembly code instruction that is to access

Art Unit: 2193

the parameter on the stack based on an offset from a common reference associated with the second procedure (this limitation of using a base pointer on the stack would be disclosed because that is how C/C++ stack is used according to known concept of calling function as set forth in APA, wherein the *BP* of Specifications Fig. 2 reads on the common reference).

But Muldner does not explicitly disclose a compiler to generate assembly code instructions, wherein the assembly code instructions. A compiler to generate assembly code for debugging purpose, platform architecture matching or for porting is suggested in Muldner attempt to port C++ code. And GNUCC discloses compiler option to create assembly instructions and passing compiler instructions/options to this assembler (Part 2: Output to Assembler Language, pg. 51-65). It would have been obvious for one skill in the art to provide a compiler generating assembly instruction codes as purported by GNUCC in light of the porting endeavor by Muldner, because implementing target code toward an architecture specific machine language can be visualized via assembly language as this can support Muldner's compiler to utilize its debugging options so to correct issues relevant to any target architecture via this step of using an assembler.

As per claims 16-17, see APA in light of Muldner's C/C++ language and rejection of claim 3.

As per claim 18, in light of claim 15 in regard to compiler for assembly instructions, see Muldner (pg. 1-2, pg.16), wherein the first assembly code instruction is to define the variable.

As per claim 26, this claim corresponds to claim 6, hence is rejected with the same rationale as set forth therein.

Conclusion

Art Unit: 2193

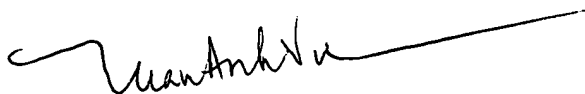
14. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (272) 272-3735. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (571)272-3719.

The fax phone number for the organization where this application or proceeding is assigned is (571) 273-3735 (for non-official correspondence – please consult Examiner before using) or 571-273-8300 (for official correspondence) or redirected to customer service at 571-272-3609.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).



Tuan A Vu
Patent Examiner,
Art Unit 2193
September 18, 2006